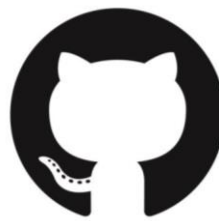




git



GitHub

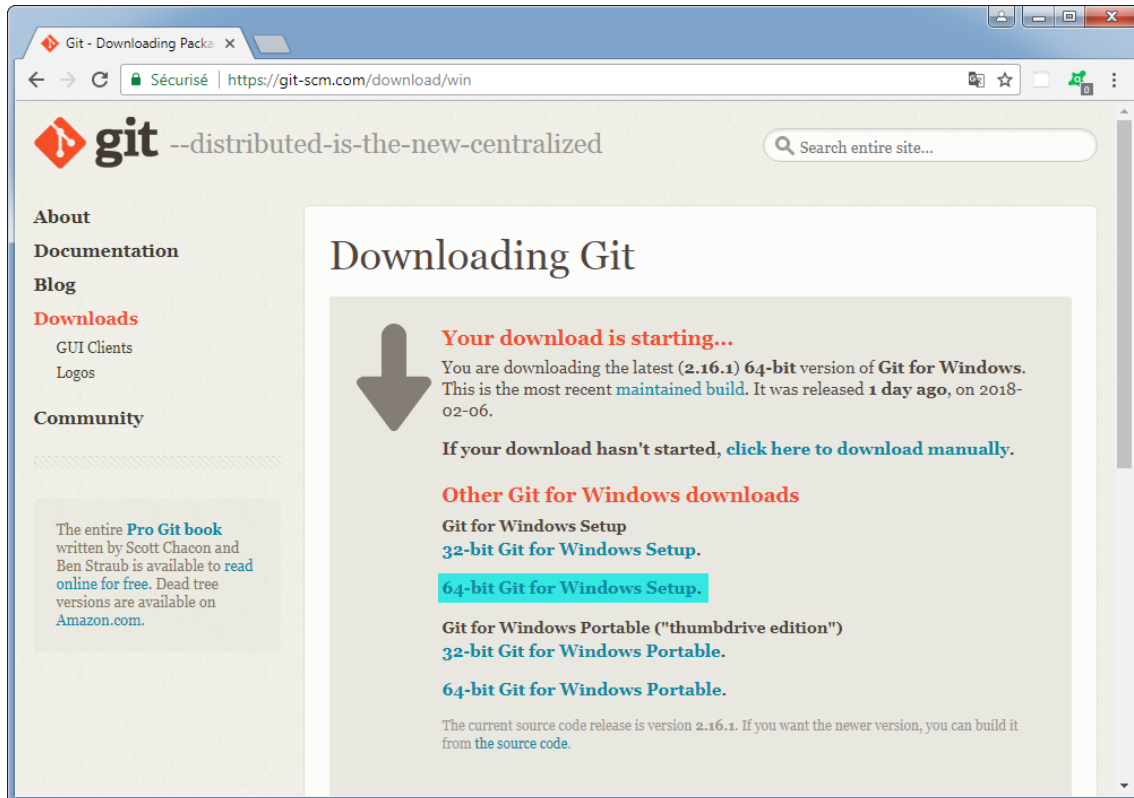
Prérequis

- RAM : Aucun prérequis.
- Espace disque : Aucun prérequis.
 - Provisionner 5 Go de disque pour un projet de taille moyenne.
- Système d'exploitation : Disponible sous Windows, Linux/Unix, MAC OS X.
 - Il existe une adhérence entre la version de Git et le système d'exploitation.

1. Installation de Git

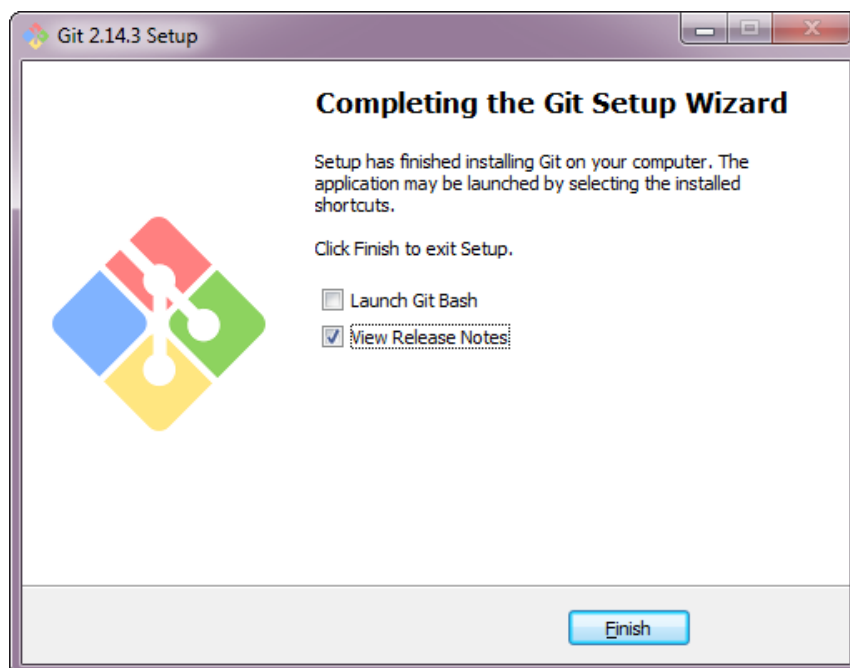
Etape 1: Téléchargement de Git

- Télécharger Git depuis son site web officiel <https://git-scm.com/download/win>



Etape 2: Installation de Git

- Exécuter l'installateur Git et choisir une installation standard.



2. Création d'un dépôt central

- Lancer l'outil Git Bash via :

Menu Démarrer > Tous les programmes > Git > Git Bash

- Créer et naviguer dans le répertoire dans le quel sera créé le dépôt central du projet. L'option `--bare` permet d'initialiser le dépôt sans un répertoire de travail. Par convention, le référentiel doit être nommé `.git` :

```
mkdir -p /c/integ_continue/git/training-app
cd /c/integ_continue/git/training-app
```

- Lancer la commande d'initialisation du dépôt:

```
git init --bare /c/integ_continue/git/training-app/.git
```

- Le répertoire `.git` contient les métadonnées du dépôt.

```

MINGW64:/c/integ_continue/git/training-app

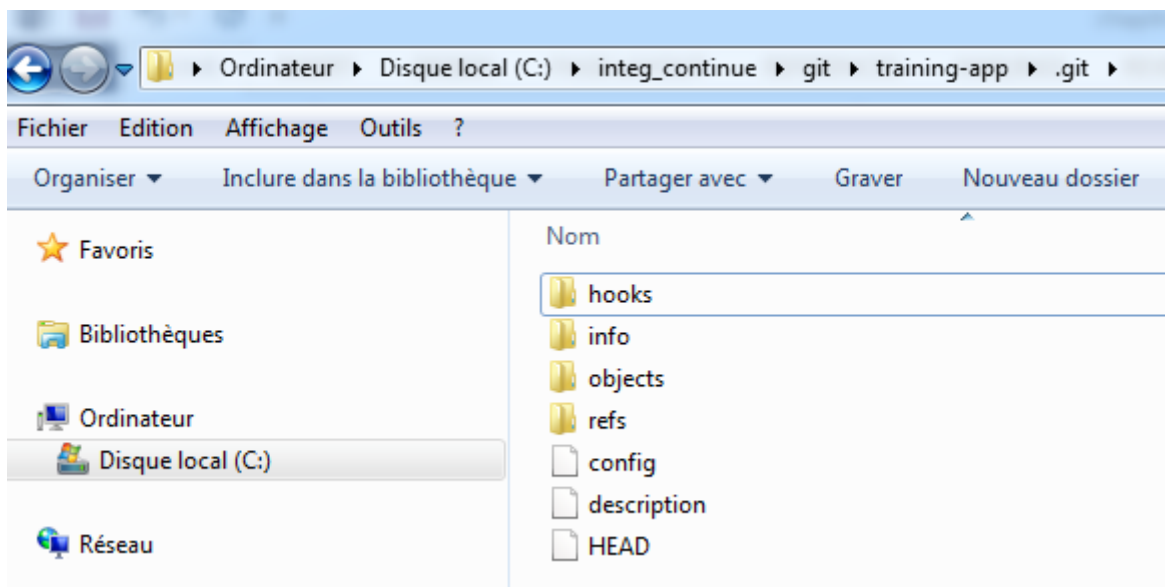
walid@walid-PC MINGW64 ~
$ mkdir -p /c/integ_continue/git/training-app

walid@walid-PC MINGW64 ~
$ cd /c/integ_continue/git/training-app/

walid@walid-PC MINGW64 /c/integ_continue/git/training-app
$ git init --bare /c/integ_continue/git/training-app/.git
Initialized empty Git repository in C:/integ_continue/git/training-app/.git/

walid@walid-PC MINGW64 /c/integ_continue/git/training-app (master)
$ |

```



3. Création d'un dépôt local

- Lancer l'outil Git Bash via :

Menu Démarrer > Tous les programmes > Git > Git Bash

- Créer et naviguer dans le répertoire dans le quel sera créé le dépôt local du projet :

```
mkdir -p /c/integ_continue/dev/training-app
cd /c/integ_continue/dev/training-app
```

- Lancer la commande d'initialisation du dépôt:

```
git init
```

```

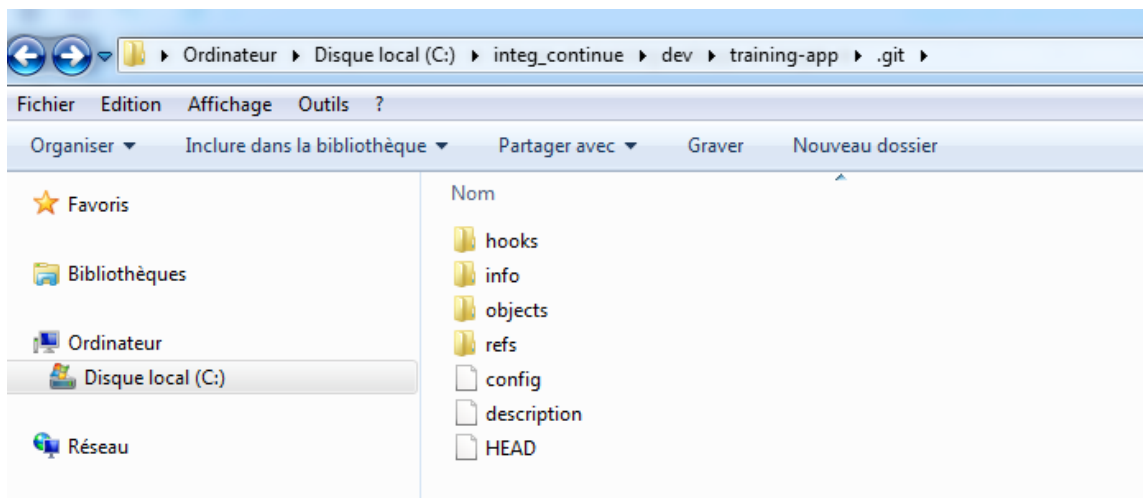
MINGW64:/c/integ_continue/dev/training-app

walid@walid-PC MINGW64 /c
$ mkdir -p /c/integ_continue/dev/training-app

walid@walid-PC MINGW64 /c
$ cd /c/integ_continue/dev/training-app

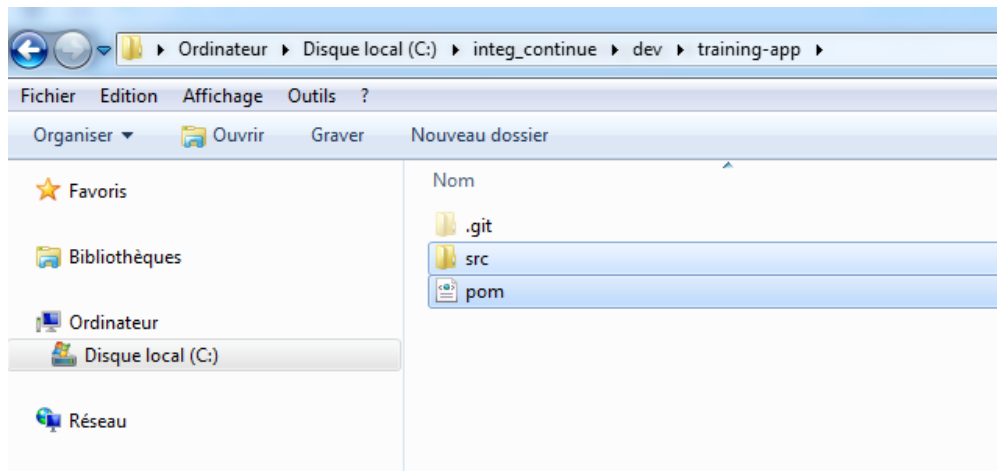
walid@walid-PC MINGW64 /c/integ_continue/dev/training-app
$ git init
Initialized empty Git repository in C:/integ_continue/dev/training-app/.git/

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ |
  
```



- Ajouter les sources du projet Maven (C:/Maven/training-app/) dans le répertoire:

/c/integ_continue/dev/training-app



- Ajouter les fichiers du projet sous le contrôle de version Git en exécutant la commande :

git add .

- Consigner définitivement les fichiers au contrôle de version Git en exécutant la commande:

git commit -m 'initialisation du depot'

```
MINGW64:/c/integ_continue/dev/training-app

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git add .

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git commit -m 'initialisation du depot'

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: empty ident name (for <walid.saadd@gmail.com>) not allowed

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git config --global user.name "wsaad"

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git config --global user.email "walid.saadd@gmail.com"

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git add .

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git commit -m 'initialisation du depot'
[master (root-commit) a6e93c3] initialisation du depot
3 files changed, 217 insertions(+)
create mode 100644 pom.xml
create mode 100644 src/main/java/com/mycompany/app/App.java
create mode 100644 src/test/java/com/mycompany/app/AppTest.java

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
```

4. Synchronisation du dépôt central avec le dépôt local

- Ajouter le lien vers le dépôt central dans le dépôt local avec la commande (à exécuter une seule fois) :

```
git remote add origin /c/integ_continue/git/training-app
```

➔ NB :s'il y a eu une erreur de path lors de l'association il faut taper la commande : **git remote remove origin**

- Publier les fichiers ajoutés dans le dépôt local vers le dépôt central avec la commande :

```
git push origin master
```

➔ *Origin* est le nom du dépôt central, *master* est le nom de la branche.

NB : si le username et l'email n'existent pas dans la config de Git il faut les ajouté en tapant :

```
git config --global user.name "Mona Lisa"
```

```
git config --global user.email monemail@domaine.com
```

Essayer les commandes : git status, git rm, git log, etc.

```

MINGW64:/c/integ_continue/dev/training-app

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git remote add origin /c/integ_continue/git/training-app

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git log
commit a6e93c327adb4e1432c726e6a762c435df94e3aa (HEAD -> master)
Author: wsaad <walid.saadd@gmail.com>
Date: Thu Jun 21 22:20:42 2018 +0200

    initialisation du depot

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git remote -v
origin C:/integ_continue/git/training-app (fetch)
origin C:/integ_continue/git/training-app (push)

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git remote show origin
* remote origin
  Fetch URL: C:/integ_continue/git/training-app
  Push URL: C:/integ_continue/git/training-app
  HEAD branch: (unknown)

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git push origin master
Counting objects: 16, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (16/16), 2.75 KiB | 281.00 KiB/s, done.
Total 16 (delta 0), reused 0 (delta 0)
To C:/integ_continue/git/training-app
 * [new branch]      master -> master

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git remote show origin
* remote origin
  Fetch URL: C:/integ_continue/git/training-app
  Push URL: C:/integ_continue/git/training-app
  HEAD branch: master
  Remote branch:
    master tracked
  Local ref configured for 'git push':
    master pushes to master (up to date)

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$

```

FIN – LAB 1

5. Manipulation des branches

Tous les **VCS** ont une forme ou une autre de gestion de branche. Créer une branche signifie diverger de la ligne principale de développement et continuer à travailler sans se préoccuper de cette ligne principale.

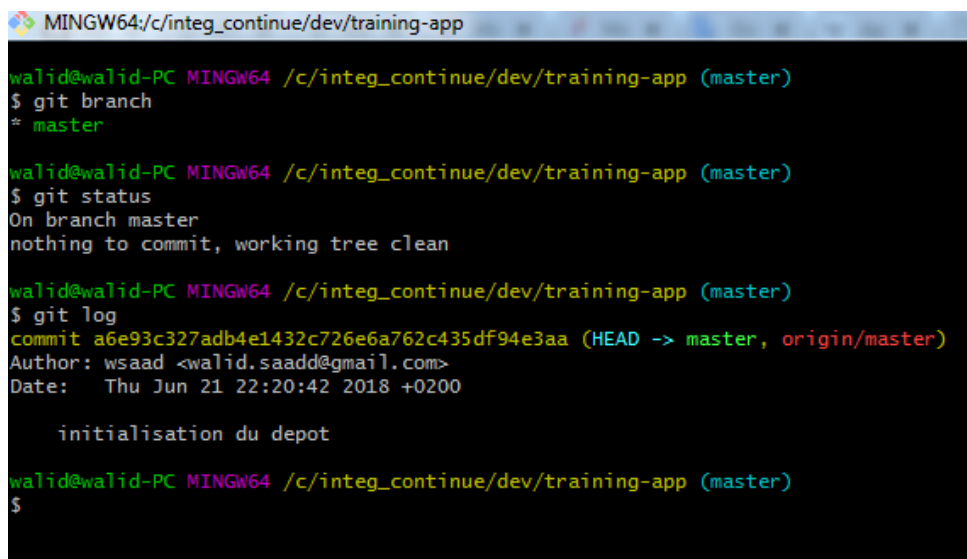
De nombreux développeurs font référence au modèle de gestion de branche de Git par rapport à la communauté des gestionnaires de version. En effet, la méthode de Git pour gérer les branches est particulièrement légère, permettant de réaliser des embranchements quasi instantanément et de basculer entre les branches généralement aussi rapidement.

À la différence de nombreux autres gestionnaires de version, Git encourage à travailler avec des méthodes qui privilégient la création et la fusion de branches, jusqu'à plusieurs fois par jour.

5.1. Concept des branches

Lorsqu'on commit dans Git, le système stocke un objet commit qui contient un **pointeur** vers l'instantané (snapshot) du contenu qui a été indexé, les méta-données d'auteur, le message et zéro ou plusieurs pointeurs vers le ou les commits qui sont les parents directs de ce commit : zéro parent pour le premier commit, un parent pour un commit normal et des parents multiples pour des commits qui sont le résultat de la fusion d'une ou plusieurs branches.

Dans notre projet, la branche par défaut sous git se nomme master cette dernière est implicite lors de la création de dépôt nous le voyons lors de l'utilisation de la commande *git status*. La branche master est simplement un pointeur (appelé **HEAD**) vers un objet commit (a6e93c327adb4e1432c726e6a762c435df94e3aa)



```

MINGW64/c/integ_continue/dev/training-app
walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git branch
* master

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git status
On branch master
nothing to commit, working tree clean

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git log
commit a6e93c327adb4e1432c726e6a762c435df94e3aa (HEAD -> master, origin/master)
Author: wsaad <walid.saadd@gmail.com>
Date: Thu Jun 21 22:20:42 2018 +0200

    initialisation du depot

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$
  
```

Afin de démontrer l'utilisation des branches le plus simple reste de réaliser une démonstration, nous allons donc créer une branche nommé **testing**.

- Créer une nouvelle branche (le * indique sur quelle branche nous travaillons actuellement sur notre dépôt local) :

```
MINGW64:/c/integ_continue/dev/training-app

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git branch testing

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git branch
* master
  testing

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$
```

- Basculer vers la nouvelle branche testing et effectuer un commit:

```
MINGW64:/c/integ_continue/dev/training-app

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git checkout testing
Switched to branch 'testing'

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git branch
  master
* testing

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git status
On branch testing
nothing to commit, working tree clean

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git log
commit a6e93c327adb4e1432c726e6a762c435df94e3aa (HEAD -> testing, origin/master, master)
Author: wsaad <walid.saadd@gmail.com>
Date: Thu Jun 21 22:20:42 2018 +0200

    initialisation du depot

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ |
```

```

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git add README.txt

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git commit -a -m 'Ajout du fichier README'
[testing 6dbd682] Ajout du fichier README
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.txt

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git status
On branch testing
nothing to commit, working tree clean

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git log
commit 6dbd682a6535e90632a925b031be28ebe2c6aaef (HEAD -> testing)
Author: wsaad <walid.saadd@gmail.com>
Date: Fri Jun 22 01:14:37 2018 +0200

    Ajout du fichier README

commit a6e93c327adb4e1432c726e6a762c435df94e3aa (origin/master, master)
Author: wsaad <walid.saadd@gmail.com>
Date: Thu Jun 21 22:20:42 2018 +0200

    initialisation du depot

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$

```

- Changer vers la branche master et effectuer un *merge* de deux branches:

```
MINGW64:/c/integ_continue/dev/training-app
walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (testing)
$ git checkout master
Switched to branch 'master'

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git status
On branch master
nothing to commit, working tree clean

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git log
commit a6e93c327adb4e1432c726e6a762c435df94e3aa (HEAD -> master, origin/master)
Author: wsaad <walid.saadd@gmail.com>
Date: Thu Jun 21 22:20:42 2018 +0200

    initialisation du depot

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git merge testing
Updating a6e93c3..6dbd682
Fast-forward
 README.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 README.txt

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git status
On branch master
nothing to commit, working tree clean

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app (master)
$ git log
commit 6dbd682a6535e90632a925b031be28ebe2c6aaef (HEAD -> master, testing)
Author: wsaad <walid.saadd@gmail.com>
Date: Fri Jun 22 01:14:37 2018 +0200

    Ajout du fichier README

commit a6e93c327adb4e1432c726e6a762c435df94e3aa (origin/master)
Author: wsaad <walid.saadd@gmail.com>
Date: Thu Jun 21 22:20:42 2018 +0200

    initialisation du depot
```

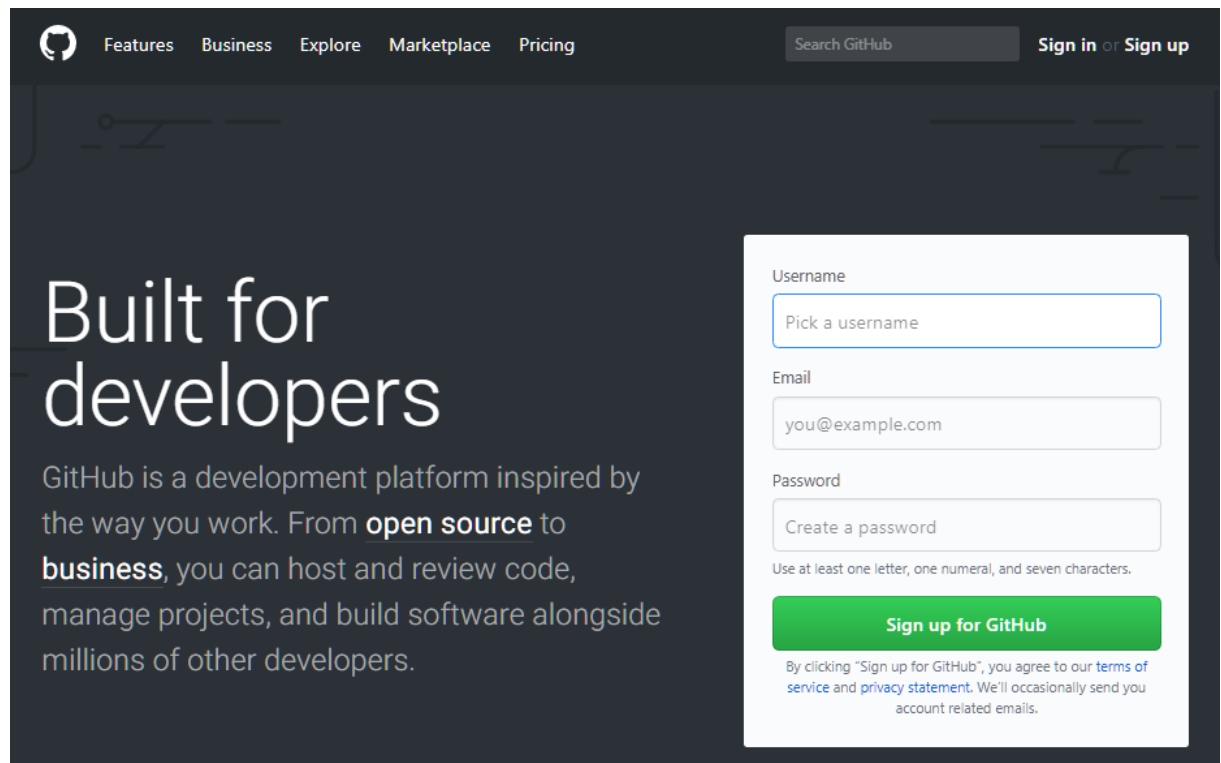
6. Travailler avec un dépôt distant public

Dans cette section, nous allons utiliser git de manière décentralisée à travers la solution [Github](#). L'utilisation d'un serveur git permet d'échanger les fichiers et offrir l'accès à tous au dépôt. La gratuité de Github s'applique uniquement pour les dépôts publics.

Pour s'authentifier auprès du serveur **github**, deux modes sont possibles avec **GIT** :

- Login / mot de passe
- RSA Key de SSH (une paire de clés publique/privée à générer par la commande **ssh-keygen**)

6.1. Créer un compte Github



The screenshot shows the GitHub homepage with a sign-up form on the right. The form includes fields for Username, Email, and Password, along with a 'Sign up for GitHub' button. The background text on the left reads: 'Built for developers. GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside millions of other developers.'

Username
Pick a username

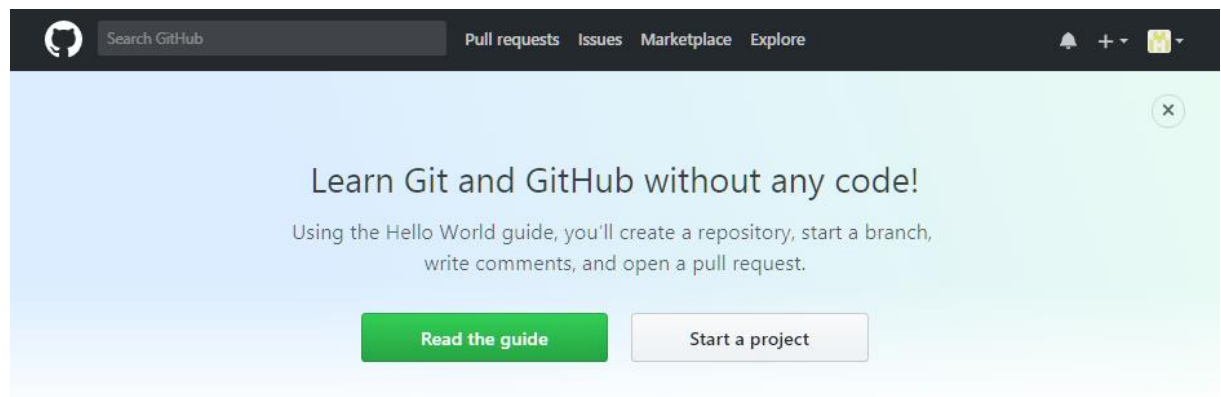
Email
you@example.com

Password
Create a password
Use at least one letter, one numeral, and seven characters.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy statement](#). We'll occasionally send you account related emails.

6.2. Créer un nouveau projet (repository)



The screenshot shows a banner on the GitHub website. The banner text reads: 'Learn Git and GitHub without any code! Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.' Below the text are two buttons: 'Read the guide' and 'Start a project'.

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Read the guide Start a project

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name

walidsaad ▼

 /

training-app ✓

Great repository names are short and memorable. Need inspiration? How about **verbose-octo-guacamole**.

Description (optional)

Mon premier dépôt Maven

☒ Public

Anyone can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

☐ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: Maven ▼

Add a license: None ▼

Create repository

.gitignore est la solution pour git afin de spécifier les fichiers qui devraient être exclus du versioning. Github propose la création automatique du fichier .gitignore de votre projet en fonction du langage utilisé (Maven ici).

walidsaad / training-app Unwatch 1 Star 0 Fork 0

[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#) [Settings](#)

Mon premier dépôt Maven Edit

[Add topics](#)

1 commit

1 branch

0 releases

1 contributor

Branch: master ▼

New pull request

Create new file

Upload files

Find file

Clone or download ▼

walidsaad Initial commit

Latest commit 5f82659 an hour from now

.gitignore

Initial commit

just now

Help people interested in this repository understand your project by adding a README. Add a README

6.3. La commande git clone

Cette commande permet de cloner en local le dépôt training-app depuis Github

```

MINGW64:/c/integ_continue/dev/training-app-github

walid@walid-PC MINGW64 ~
$ cd /c/integ_continue/dev/

walid@walid-PC MINGW64 /c/integ_continue/dev
$ git clone https://github.com/walidsaad/training-app.git training-app-github
Cloning into 'training-app-github'...
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.

walid@walid-PC MINGW64 /c/integ_continue/dev
$ cd training-app-github/

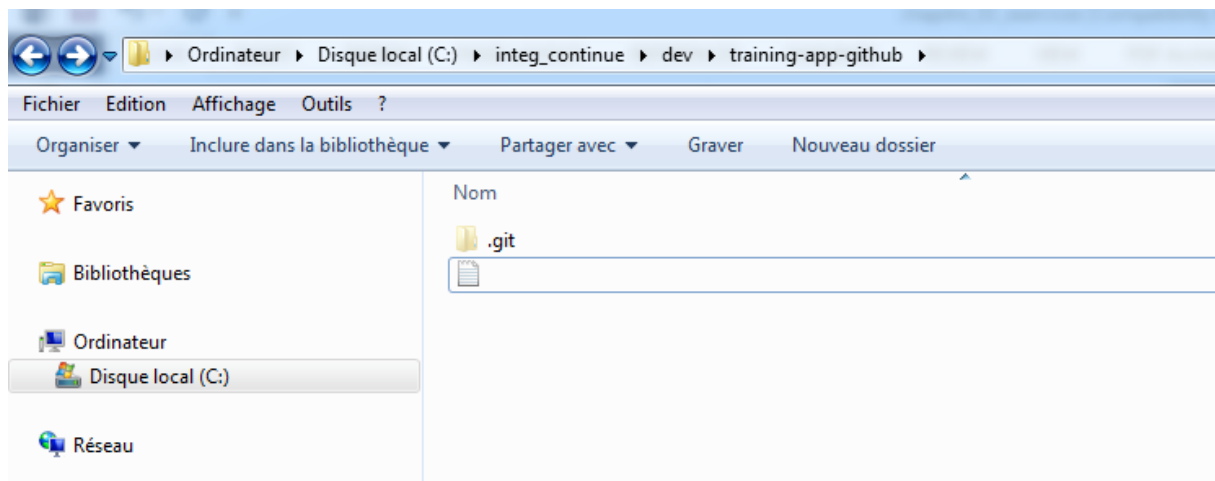
walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git branch
* master

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$

```



6.4. Synchronisation du dépôt central avec le dépôt distant Github (git remote)

D'abord, on doit visualiser le dépôt distant à configurer.

```

MINGW64:/c/integ_continue/dev/training-app-github

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git remote -v
origin https://github.com/walidsaad/training-app.git (fetch)
origin https://github.com/walidsaad/training-app.git (push)

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git remote show origin
* remote origin
Fetch URL: https://github.com/walidsaad/training-app.git
Push URL: https://github.com/walidsaad/training-app.git
HEAD branch: master
Remote branch:
  master tracked
Local branch configured for 'git pull':
  master merges with remote master
Local ref configured for 'git push':
  master pushes to master (up to date)

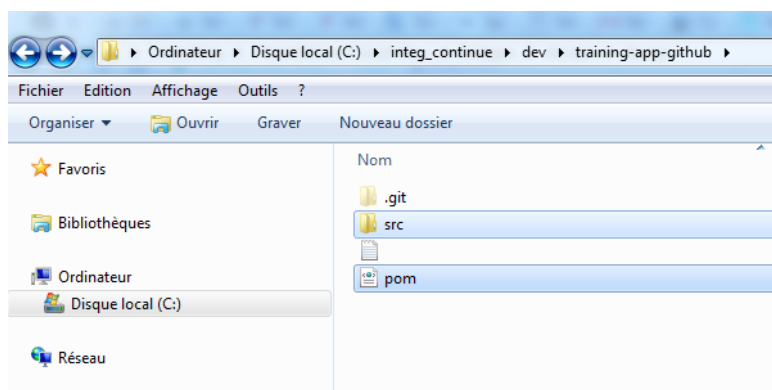
walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$

```

- **origin** : qui est notre dépôt public sur internet où tout le monde peut visualiser notre travail
- c'est la même configuration pour extraire (**fetch**) et pousser (**push**) les fichiers
- On a une seule branche **master**

6.5. Mise à jour des fichiers vers le serveur Github (push)

On veut modifier notre dépôt local par l'ajout du projet Maven. Ensuite on met à jour notre dépôt Github.



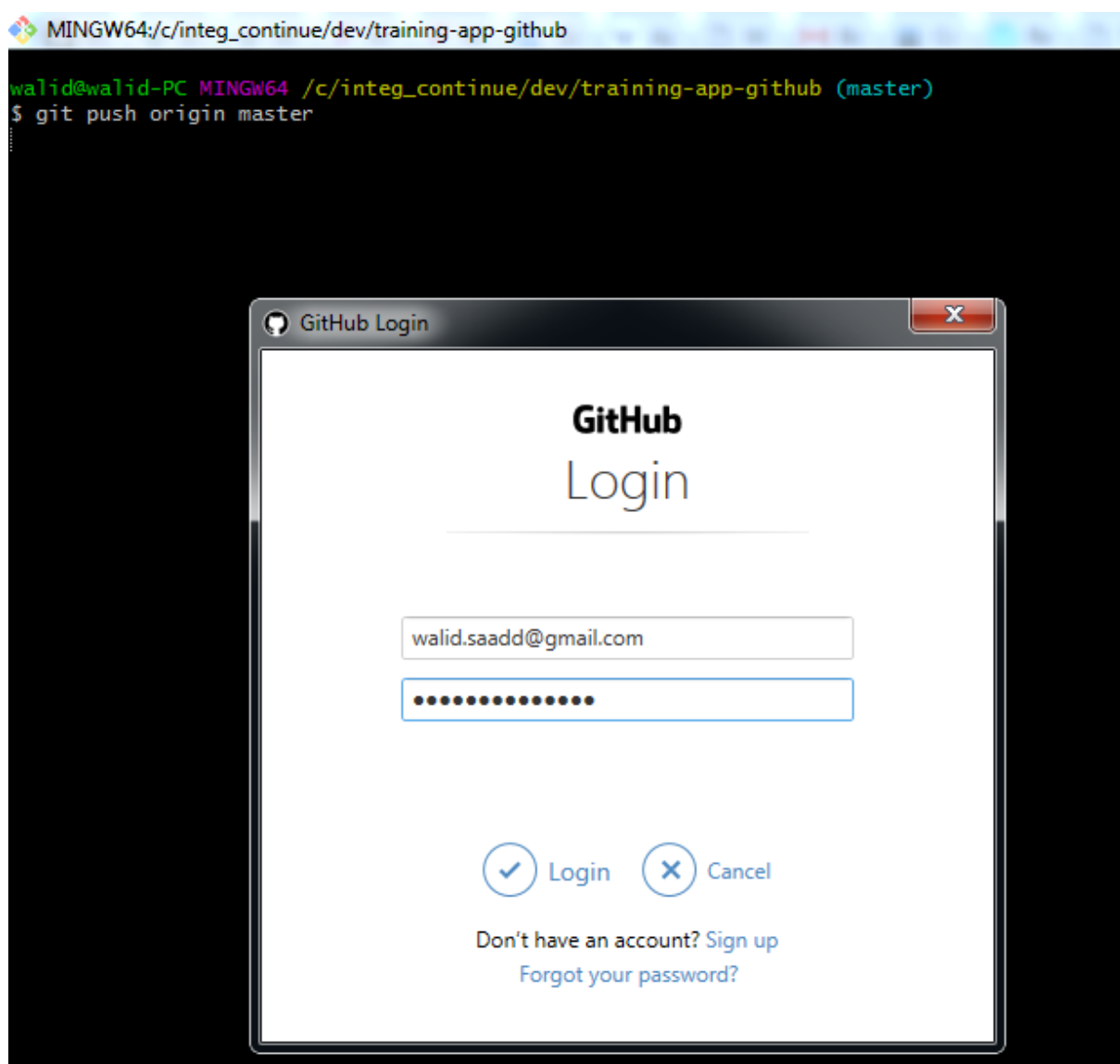

```
MINGW64:/c/integ_continue/dev/training-app-github

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git add .

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git commit -a -m 'Ajout des fichiers Maven'
[master c8825dd] Ajout des fichiers Maven
3 files changed, 223 insertions(+)
create mode 100644 pom.xml
create mode 100644 src/main/java/com/mycompany/app/App.java
create mode 100644 src/test/java/com/mycompany/app/AppTest.java

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ |
```

Lancer la commande *git push*



```

MINGW64:/c/integ_continue/dev/training-app-github
walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git push origin master
Counting objects: 16, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (16/16), 2.88 KiB | 328.00 KiB/s, done.
Total 16 (delta 0), reused 0 (delta 0)
To https://github.com/walidsaad/training-app.git
5f82659..c8825dd master -> master

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ |

```

Vérification sur Github

walidsaad / training-app

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Mon premier dépôt Maven [Add topics](#) [Edit](#)

2 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

File	Commit Message	Time
src	Ajout des fichiers Maven	16 minutes ago
.gitignore	Initial commit	an hour ago
pom.xml	Ajout des fichiers Maven	16 minutes ago

Help people interested in this repository understand your project by adding a README. [Add a README](#)

6.6. Mise à jour des fichiers depuis le serveur Github (pull)

En pratique, plusieurs utilisateurs (collaborateurs) peuvent travailler sur le même dépôt et pousser leurs modifications (ajout des fichiers, suppression, etc.). Bien entendu on veut récupérer les nouveaux fichiers. Pour ce faire nous utilisons la commande **git pull**. On doit donner la source et la branche qu'on désire récupérer.

On commence d'abord par ajouter le fichier README.md, puis commiter.

LE GESTIONNAIRE DE SOURCES (SCM)

walidsaad / training-app

Unwatch 1

Star 0

Fork 0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights

Settings


training-app / or cancel

<> Edit new file

Preview

Spaces 2 No wrap

```
1 # training-app
2 Mon premier dépôt Maven
3
```



Commit new file

Create README.md

Ajout du fichier README

☒ Commit directly to the master branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit new file

Cancel

3 commits

1 branch

0 releases

1 contributor

Branch: master


New pull request

Create new file

Upload files

Find file

Clone or download

 walidsaad Create README.md Latest commit 96958a4 an hour from now

src

Ajout des fichiers Maven

2 hours ago

.gitignore

Initial commit

3 hours ago

README.md

[Create README.md](#)

just now

pom.xml

Ajout des fichiers M

2 hours ago

README.md

Create README.md

Ajout du fichier README

Lancer commande ***git pull origin master***

```

MINGW64:/c/integ_continue/dev/training-app-github

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git pull origin master
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/walidsaad/training-app
 * branch      master      -> FETCH_HEAD
    c8825dd..96958a4  master  -> origin/master
Updating c8825dd..96958a4
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 README.md

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ git log
commit 96958a4ebdc64345113708864b4fa32ca6c93387 (HEAD -> master, origin/master, origin/HEAD)
Author: walidsaad <walid.saadd@gmail.com>
Date:   Sat Jun 23 00:05:14 2018 +0200

    Create README.md

    Ajout du fichier README

commit c8825dd53574972fc723e5a8e19b6bf3eb7039de
Author: wsaad <walid.saadd@gmail.com>
Date:   Fri Jun 22 20:57:02 2018 +0200

    Ajout des fichiers Maven

commit 5f82659367270137dd1204d0db3f8bc0fa8a6e2d
Author: walidsaad <walid.saadd@gmail.com>
Date:   Fri Jun 22 20:13:31 2018 +0200

    Initial commit

walid@walid-PC MINGW64 /c/integ_continue/dev/training-app-github (master)
$ |

```

- Le système analyse l'identifiant **HEAD** du dépôt local pour la branche **master**, l'id est **c8825ddxx**
- Il compare cette identifiant avec celui extrait depuis le serveur **origin/master**, l'identifiant est **96958a4xx**. Comme ceci n'est pas identique il débute le processus de fusion.
- Le message **Fast-forward** indique qu'il y a un commit commun dans l'historique **c8825ddxx** et que les **commits** suivants se sont ajouter avec le temps.
- Le système va simplement télécharger le nouveau dépôt et modifier le pointeur **HEAD** sur la nouvelle tête.
- On trouve aussi une description des modifications réalisé sur les fichiers, dans le cas présent uniquement le fichier **README** fut modifié.

Donc si on résume les étapes :

1. On met à jour notre dépôt local avec la commande ***git pull***
2. On fait les modifications (ajout, suppression, modifications des fichiers)
3. On fait le commit local (***git commit -a -m "Description"***)
4. On pousse nos modifications sur le serveur avec la commande ***git push***

7. Intégration Maven Github

Une fois que nous avons enregistré les modifications effectuées sur notre projet Maven dans le dépôt Github. On veut maintenant le compiler, déployer et le tester en local. Pour faire ceci, nous devons relier Maven à Github (notre SCM). Autrement dit, Maven lancera un pull de nos classes JAVA modifiés depuis le serveur Github.

- Dans un premier lieu, on aura besoin de configurer le SCM Github dans le fichier POM (pom.xml) en ajoutant la section *scm* et le plugin *maven-scm-plugin* :

```
<scm>

  <connection>scm:git:https://github.com/walidsaad/training-app.git</connection>

  <developerConnection>scm:git:https://github.com/walidsaad/training-
app.git</developerConnection>

  <url>https://github.com/walidsaad/training-app</url>

  <tag>master</tag>

</scm>

.....

<plugin>

  <groupId>org.apache.maven.plugins</groupId>

  <artifactId>maven-scm-plugin</artifactId>

  <version>1.10.0</version>

  <configuration>

    <goals>install</goals>

  </configuration>

</plugin>
```

- Dans un second lieu, lancer la commande *mvn clean* et *mvn scm:bootstrap* et vérifier que Maven a bien déployé notre projet disponible sous le serveur Github (en faisant un pull depuis Github).
- La commande *mvn scm:bootstrap* permet de faire un checkout et build du projet.

```

C:\Maven\training-app>mvn clean
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.mycompany.app:training-app >-----
[INFO] Building training-app 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ training-app ---
[INFO] Deleting C:\Maven\training-app\target
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 0.657 s
[INFO] Finished at: 2018-06-23T01:00:25+02:00
[INFO]
-----

C:\Maven\training-app>mvn scm:bootstrap
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.mycompany.app:training-app >-----
[INFO] Building training-app 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-scm-plugin:1.10.0:bootstrap (default-cli) @ training-app ---
[INFO] Removing C:\Maven\training-app\target\checkout
[INFO] Executing: cmd.exe /X /C "git clone https://github.com/walidsaad/training-app.git C:\Maven\training-app\target\checkout"
[INFO] Working directory: C:\Maven\training-app\target
[INFO] Executing: cmd.exe /X /C "git ls-remote https://github.com/walidsaad/training-app.git"
[INFO] Working directory: C:\Users\walid\AppData\Local\Temp
[INFO] Executing: cmd.exe /X /C "git pull https://github.com/walidsaad/training-app.git master"
[INFO] Working directory: C:\Maven\training-app\target\checkout
[INFO] Executing: cmd.exe /X /C "git checkout"
[INFO] Working directory: C:\Maven\training-app\target\checkout
[INFO] Executing: cmd.exe /X /C "git ls-files"
[INFO] Working directory: C:\Maven\training-app\target\checkout

[INFO] Installing C:\Maven\training-app\target\checkout\target\training-app-1.0-SNAPSHOT.jar to C:\Users\walid\.m2\repository\com\mycompany\app\training-app\1.0-SNAPSHOT\training-app-1.0-SNAPSHOT.jar
[INFO] Installing C:\Maven\training-app\target\checkout\pom.xml to C:\Users\walid\.m2\repository\com\mycompany\app\training-app\1.0-SNAPSHOT\training-app-1.0-SNAPSHOT.pom
[INFO] Installing C:\Maven\training-app\target\checkout\target\training-app-1.0-SNAPSHOT-jar-with-dependencies.jar to C:\Users\walid\.m2\repository\com\mycompany\app\training-app\1.0-SNAPSHOT\training-app-1.0-SNAPSHOT-jar-with-dependencies.jar
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 7.248 s
[INFO] Finished at: 2018-06-23T01:03:51+02:00
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 20.052 s
[INFO] Finished at: 2018-06-23T01:03:51+02:00
[INFO]
-----

C:\Maven\training-app>java -jar target\checkout\target\training-app-1.0-SNAPSHOT-jar-with-dependencies.jar
----- Connexion au serveur de données MySQL -----
Le driver JDBC pour MySQL est disponible.
Connexion à la base de données à l'adresse spécifiée avec succès.
----- Afficher toutes les sessions de formations -----
Formation Integration Continue, Maven, Toulouse, 2018-06-25, 10, 1
Formation Integration Continue, Jenkins, Toulouse, 2018-06-27, 10, 1

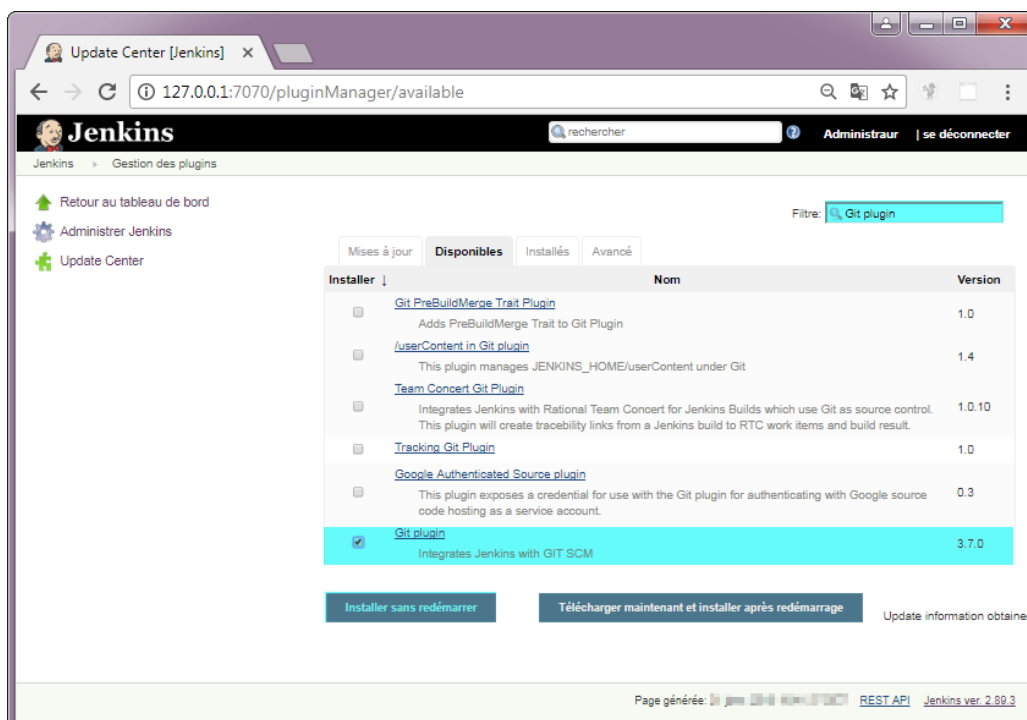
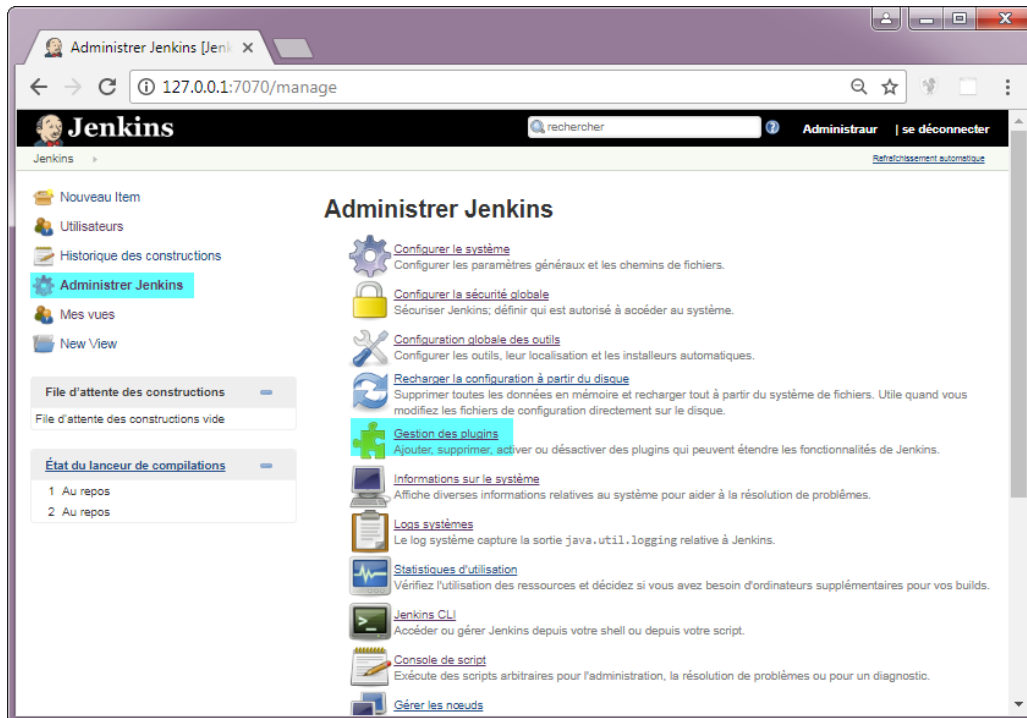
C:\Maven\training-app>_

```

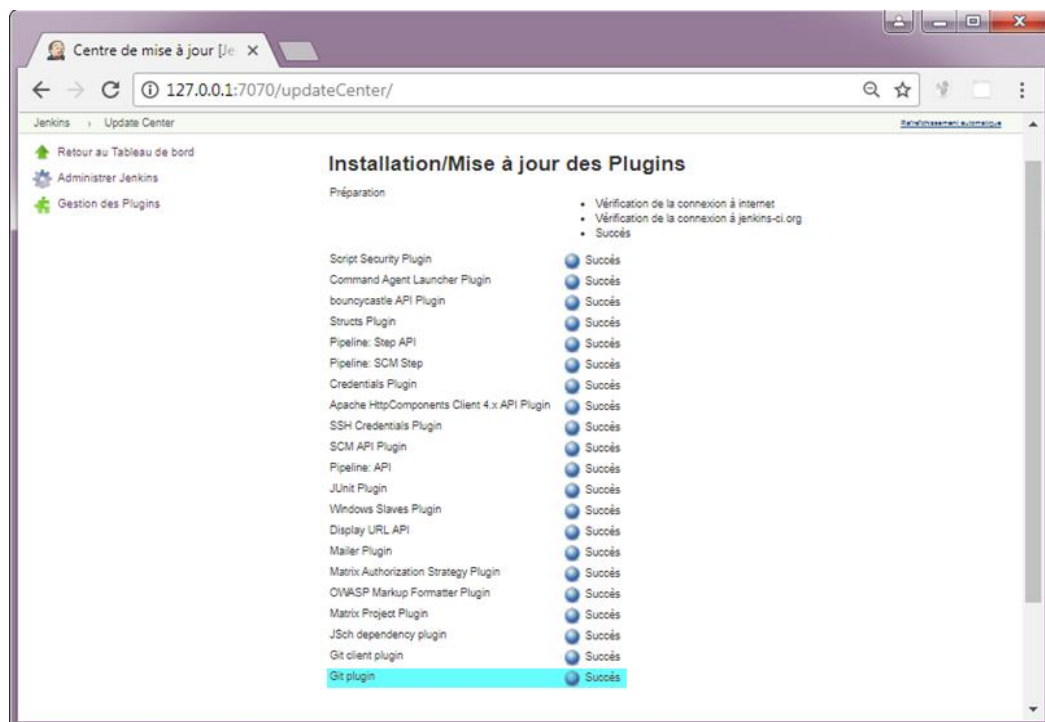
8. Intégration Git dans Jenkins

Etape 1: Installation du plugin Git

- Naviguer vers la vue "Administrer Jenkins > Gestion des plugins" et cliquer sur l'onglet "Disponibles" et filtrer avec le terme "Git plugin".

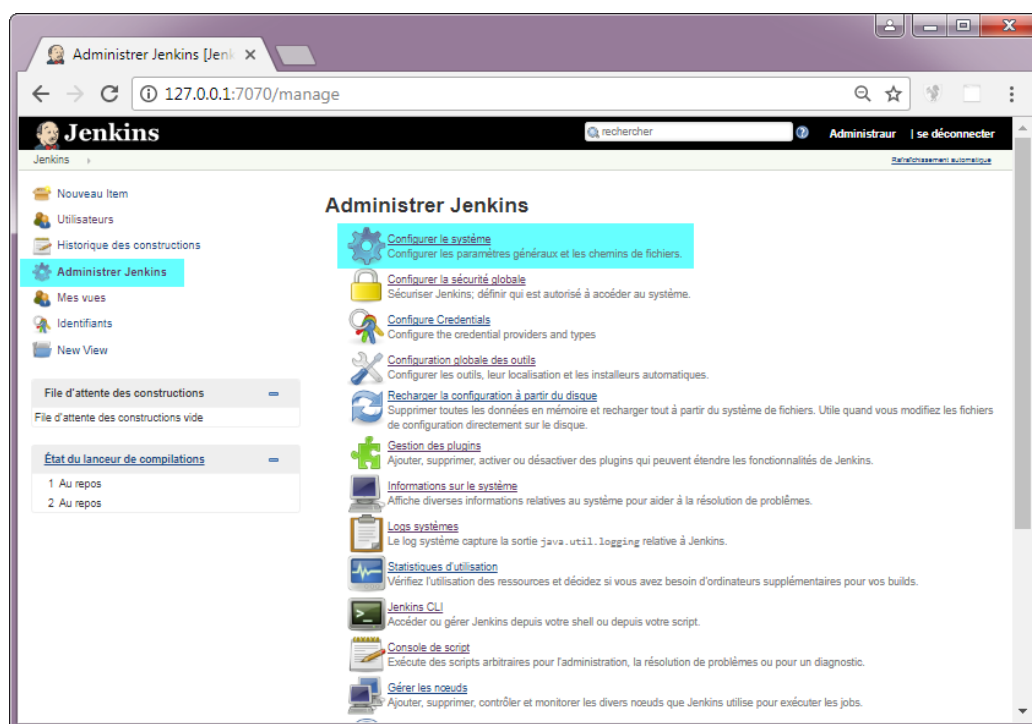


- Sélectionner l'option Git plugin et cliquer sur le bouton "Installer sans redémarrer".



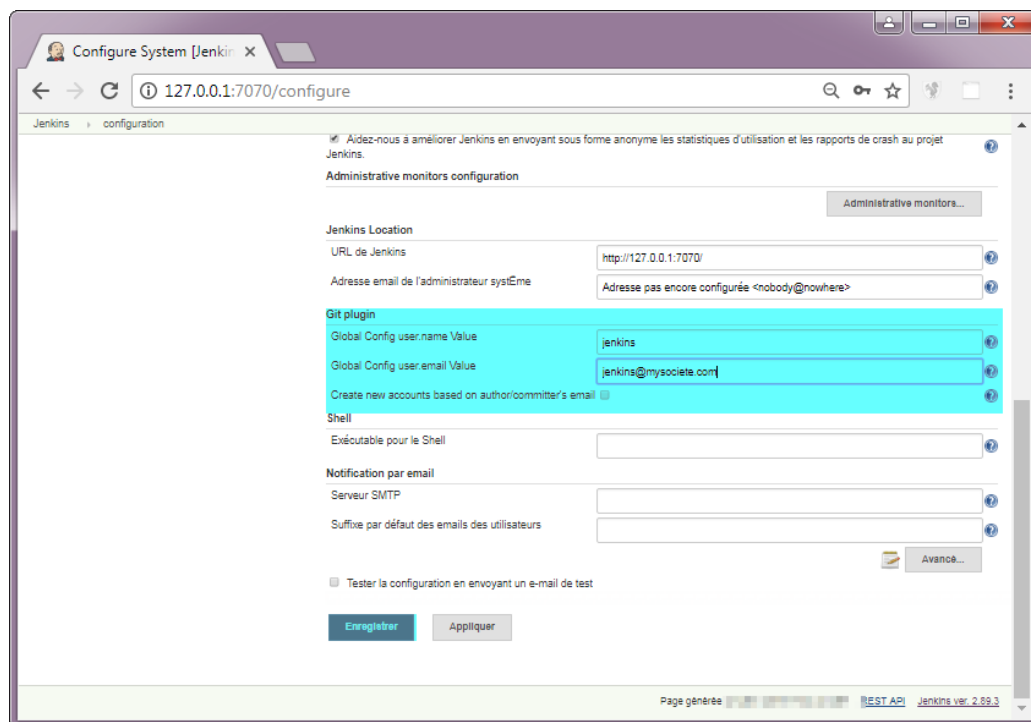
Etape 3: Configuration du plugin Git

- Naviguer vers la vue "Administrer Jenkins > Configurer le système" et se positionner sur la rubrique "Git plugin".

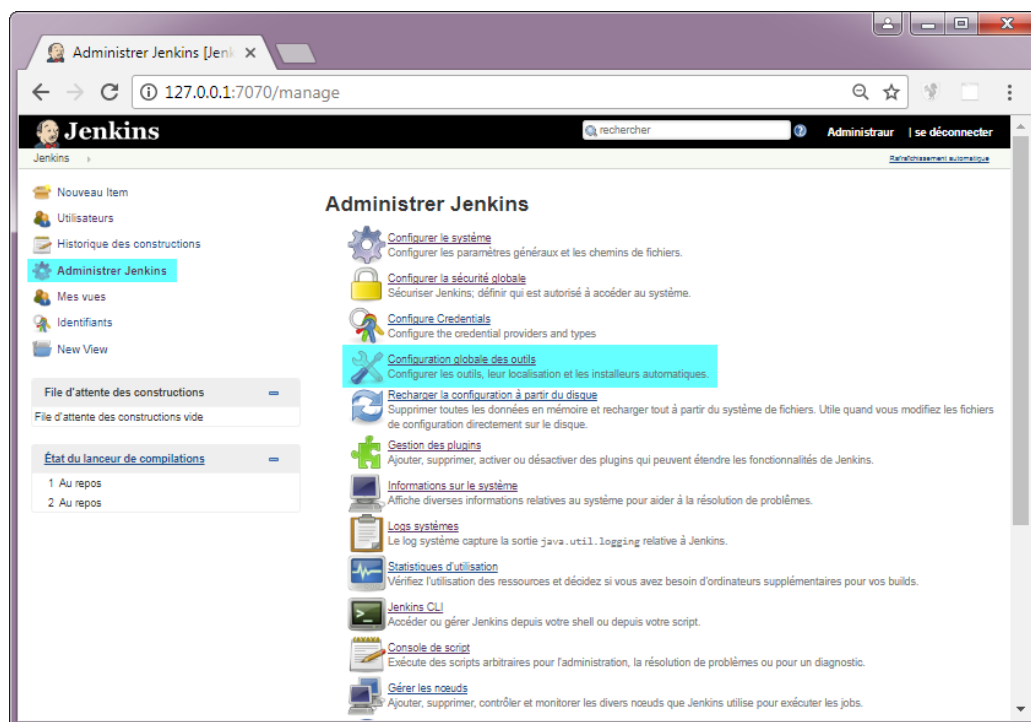


LE GESTIONNAIRE DE SOURCES (SCM)

- Saisir le nom de l'utilisateur Git ainsi que son adresse email. Cet utilisateur sera utilisé par Jenkins pour récupérer le code source depuis Git. Et puis cliquer sur Enregistrer.



- Naviguer vers la vue "Administrer Jenkins > Configuration globale des outils" et se positionner sur la rubrique "Git".



LE GESTIONNAIRE DE SOURCES (SCM)

- Saisir le nom de Git et son chemin d'installation dans les champs "Name" et "Path to Git executable" et par la suite cliquer sur le bouton "Enregistrer".

